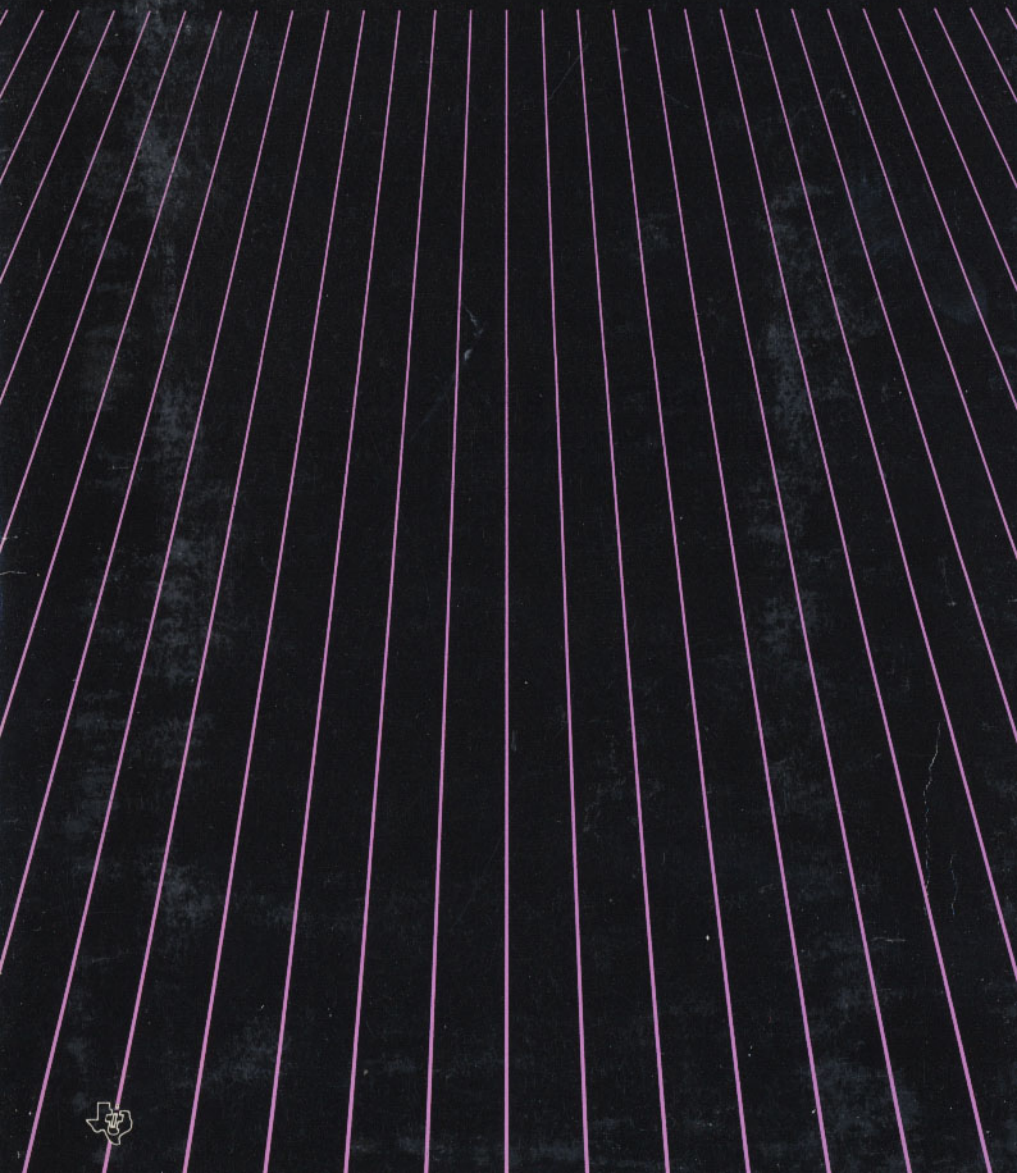
TEXAS INSTRUMENTS **RS232**
USERS MANUAL



QUICK REFERENCE GUIDE

SOFTWARE OPTION ENTRIES (RS232 PORT)

OPTION	ENTER AS
Baud rate	B = (50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200)
Data bits	D = (5, 6, 7, 8)
Parity	P = (O, E, N, S, M)
Parity check	C = (N, Y)
Nulls	N = (0-99)
Stop bits	S = (1, 2)
Echo	E = (N, Y)
Transfer type	T = (R, C, W)
Data overrun	O = (N, Y)
Carriage return	R = (N, C, L)

SOFTWARE OPTION ENTRIES (PARALLEL PORT)

OPTION	ENTER AS
Carriage return	R = (N, C, L)
Strobe level	S = (N, P)

TEXAS INSTRUMENTS RS232 USERS MANUAL

IMPORTANT

Record the serial number from the label on the unit and the purchase date in the space below. The serial number is identified by the words "SER. NO." printed on the label. Always reference this information in any correspondence.

HX-3000		
Model No.	**Serial No.**	**Purchase Date**

TABLE OF CONTENTS

Introduction 1
Set-Up Instructions 2
  Connecting the RS232 Peripheral 3
  Testing the Peripheral 4
  Connecting Devices to the Peripheral 5
Operating the System 7
  Using BASIC 7
  Software Options 14
Recovery from Errors 21
Common Applications 22
  Output to a Printer on the Parallel Port 22
  Exchanging Messages Between a Computer and a Video Terminal 22
Appendices 24
  A. Error Codes 24
  B. Using Assembly Language 26
  C. Parity Options 31
  D. RS232 Cabling 32
  E. Parallel Port Cabling 34
Service Information 35
  In Case of Difficulty 35
  Exchange Centers 36
  If You Have Questions or Need Assistance 36
Warranty 38

INTRODUCTION

The HX-3000 RS232 peripheral is a communications adapter that broadens the range of devices you can attach to computers that use the TI *HEX-BUS*™ Intelligent Peripheral Interface. This peripheral offers two versatile means of connecting devices: the RS232 port and the optional parallel port. The RS232 port lets you attach video display terminals, modems, plotters, and other devices which follow the EIA RS232C serial interface specification. The parallel port is available for use with many of the popular computer printers built for parallel interfacing.

As part of the expanding line of TI devices that use the *HEX-BUS* interface (a standardized interconnection system with a uniform set of cabling conventions, control signals, and message structures), the RS232 peripheral plugs directly into any computer compatible with this interface system.

This manual shows you how to connect and test the RS232 peripheral and presents the BASIC instructions that are used to operate it. A description of the software options (parameters that may be varied to configure the peripheral for communication with a variety of attached devices) follows the section on BASIC. Some common applications of the peripheral are given next, followed by a number of useful appendices and a section of service information.

Explanations and examples of programming employ the version of BASIC developed for the TI Compact Computer Model CC-40. Users of other TI computers may note differences in format between the BASIC statements shown here and equivalent statements as used with their own machines. Apart from that distinction, however, the material in this manual applies to operation of the peripheral with any computer compatible with the *HEX-BUS* interface.

SET-UP INSTRUCTIONS

Setting up the HX-3000 RS232 peripheral is a simple process. First the peripheral is attached to the *HEX-BUS*™ interface. Then its operation is tested. This section describes the steps involved in each of these procedures. Information about connecting other devices to the peripheral is also included. Please read the material in this section completely before you begin to set up the peripheral.

CAUTION

The electronic components of the RS232 peripheral can be damaged by discharges of static electricity. To avoid damage, do not touch the connector contacts or expose them to static electricity.

After you have unpacked the RS232 peripheral, you are ready to attach it to the *HEX-BUS* interface. Save the packing material for storing or transporting the device.

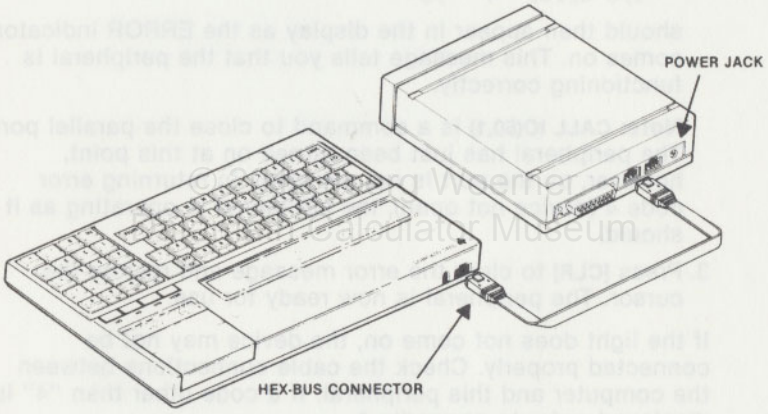
The devices in the TI *HEX-BUS*™ Intelligent Peripheral Interface system have identical eight-pin recessed connectors for the cable through which they communicate. The computer has one such connector, while each peripheral has two of them so that a series of devices may be attached to the computer. The first peripheral is plugged directly into the computer, the second peripheral is cabled to the first, and so on. The last peripheral has one connector free.

You may link devices to the computer in any order. Just plug a new section of *HEX-BUS* interface cable into the available connector and plug the other end of the cable into one of the connectors on the new peripheral, as described on the next page. The plugs are keyed so that you can insert them only one way.

Peripherals are normally arranged in a stack next to the computer, using the short sections of cable supplied with the devices. Longer cables are available separately if you prefer to arrange the peripherals differently.

Connecting the RS232 Peripheral

1. Turn off the computer.
2. If other peripherals are already attached to the *HEX-BUS*™ interface, wait for their activity to cease. Then turn them off.
3. Locate the device having the one available *HEX-BUS* connector (either the last peripheral on the bus, or the computer if no peripherals are attached yet). Holding that device firmly, plug one end of the cable into the connector.
4. Place the RS232 peripheral in position and attach the other end of the cable to either connector on this unit.



5. Attach the power cord to the small jack on the back of the RS232 peripheral and plug the AC adapter model AC9201 (included with the peripheral) into a standard 115-volt outlet. DO NOT CONNECT ANY OTHER ADAPTER TO THIS PERIPHERAL.

CAUTION

To prevent damage, disconnect all devices before moving any part of the *HEX-BUS* system. The cables and connectors which link the computer and peripherals are subject to accidental strain if not detached. For shipment over long distances repack the system securely, preferably in its original packing materials.

Testing the Peripheral

Note: The following test procedure is used with the CC-40. Computers other than the CC-40 may require different test procedures from that listed below.

1. Turn on the RS232 peripheral and any other attached peripherals first. Then turn on the computer.

   Note: All peripherals must be turned on for proper operation.

2. Type **CALL IO(50,1)** and press **[ENTER]**. The CALL IO statement causes the I/O (Input/Output) indicator in the computer display and the "working" light on the peripheral to turn on for an instant. The message

   ```
   I/O error  4  "50"
   ```

   should then appear in the display as the ERROR indicator comes on. This message tells you that the peripheral is functioning correctly.

   Note: **CALL IO(50,1)** is a command to close the parallel port. The peripheral has just been turned on at this point, however, so the port is not yet open. In returning error code 4 (device not open), the peripheral is operating as it should.

3. Press **[CLR]** to clear the error message and restore the cursor. The peripheral is now ready for use.

If the light does not come on, the device may not be connected properly. Check the cable connections between the computer and this peripheral. If a code other than "4" is displayed, refer to appendix A.

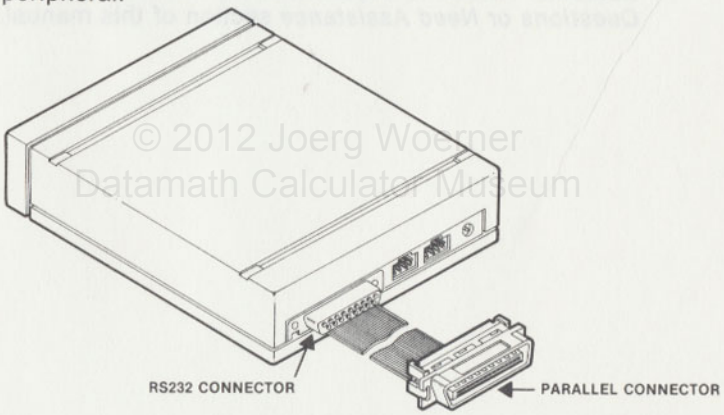
If the I/O indicator stays on, check that all peripherals are powered up. The computer cannot respond to input from the keyboard while in this state. Turn the peripheral off momentarily to clear the condition. Then check the cable connections and try the operational check once more. If the malfunction persists, see the *In Case of Difficulty* section of this manual for further assistance.

Connecting Devices to the Peripheral

After the peripheral is connected to the *HEX-BUS* interface and tested, additional devices can be attached to the RS232 and parallel ports.

The RS232 port is available for both input and output. It is readily accessible at the back of the peripheral housing, next to the two eight-pin *HEX-BUS* connectors. Just plug in the RS232 cable from the other device.

Most devices built according to the Electronic Industries Association (EIA) RS232C interface standard may be attached to the RS232 port. To be sure that a given device is compatible with the peripheral, check its user's manual for specific variations within the standard. Appendix D contains information about the RS232 cabling required by this peripheral.



The optional parallel port is provided for output only. It serves expressly as a connector for the numerous computer printers that follow the parallel port protocol used in this peripheral.

The parallel port is located inside the peripheral housing. Peripherals equipped with this option have a permanently-attached flat cable which passes beneath the RS232 connector and plugs directly to most printers.

Appendix E supplies reference information on the signal and control lines used in the cable. Check the manual of the printer you intend to connect to verify the compatibility of that device with the parallel port.

If the Texas Instruments 99/4 Impact Printer Model PHP2500 is used on the parallel port, the RS232 board in the printer must be removed. Refer to appendix D in the printer manual (*Setting DIP Switches*) for instructions on removing the RS232 board.

Note: RS232 peripherals sold without the parallel port option can be equipped with this feature after purchase as well. If you wish to have the parallel output capability added at some point, contact Texas Instruments for further information. The address and telephone numbers of the TI Customer Relations Department are found in the *If You Have Questions or Need Assistance* section of this manual.

OPERATING THE SYSTEM

After you have attached a device to the RS232 peripheral, you can control the operation of the system from the computer keyboard in one of several ways.

- You can operate peripherals with BASIC statements and commands in programs of your own, as described in this manual.
- You can use separately available *Solid State Software*™ cartridges which allow you to access peripherals by responding to prompts in prewritten programs for engineering, science, business, and financial applications.
- If you are using a CC-40 computer equipped with an Editor/Assembler cartridge, you can control peripherals in assembly language.

Appendix B gives an example of an assembly-language application utilizing the RS232 port. For further information on the use of *Solid State Software* or Editor/Assembler cartridges with the RS232 peripheral, refer to the manuals for those cartridges.

Using BASIC

The BASIC statements and commands which may be used with the RS232 peripheral are summarized below. All of them are implemented in the CC-40 computer. The CALL IO statement may not be available in other TI computers.

- LIST—Prints or displays a copy of a program.
- OPEN, CLOSE—Begin and end the use of a device in a program.
- INPUT, LINPUT, PRINT—Send and receive data. INPUT and LINPUT are used only with the RS232 port, *not* with the parallel port.
- CALL IO—Calls a subprogram which can be used to communicate with peripherals. Allows greater control over internal details of operations than is otherwise possible in BASIC.
- EOF (End-of-File)—Tests whether a data file being input contains additional records to be read. Seldom used with this peripheral.

The OPEN statement must precede all of the other statements and commands described above except for LIST and CALL IO. These two instructions may be used independently as well as after OPEN.

Statements and Commands

Statements are BASIC instructions within a program which are executed when the program is run. Commands are BASIC instructions outside a program which are executed immediately. Every instruction belongs to one of these two categories, though some overlap exists. On the next few pages the BASIC statements and commands which can be used with this peripheral are described.

The sample formats provided with each description are those used with the CC-40 computer. Other TI computers compatible with the *HEX-BUS*™ interface may use slightly different BASIC formats.

The OPEN Statement

The OPEN statement prepares a BASIC program for communication with peripheral devices. When used with this peripheral, the OPEN statement links an attached device to a file number and sets functional parameters for the operation of the RS232 or parallel port. The OPEN statement for this peripheral has the following general form.

> OPEN #*file-number*, "*device-number* [.*software-options*] "
> [, *file-attributes*]

File-number is a numeric expression that evaluates to an integer between 1 and 255.

Device-number is the numeric designation of the peripheral to be operated. Each peripheral has a number of its own. A device on the RS232 port may have any number from 20 through 23. Numbers for devices on the parallel port may range from 50 through 53.

Note: The device numbers for this peripheral are factory-set at 20 for the RS232 port and 50 for the parallel port. You never need to change them unless you use more than one RS232 peripheral on the *HEX-BUS*™ interface. If your applications do require more than one of these peripherals, write to Texas Instruments Incorporated, RS232 Device Codes, P.O. Box 53, Lubbock, Texas 79408 for information on how to alter device numbers.

Software-options are a set of parameters that you may vary to match the characteristics of this peripheral to those of attached devices. If you choose not to vary them, these parameters automatically assume certain settings called *default values.* For further information, consult the *Software Options* section of this manual.

File-attributes are optional keywords which define certain features of the data file itself. For attributes not specified, default values are assumed. These features are described below.

- File organization—SEQUENTIAL is the only file organization which may be used with the RS232 peripheral. This is the default value and need not be specified in the OPEN statement. RELATIVE (random access) files cannot be used.
- File type—It is possible for the *HEX-BUS*™ interface system to process data either in ASCII (American Standard Code for Information Interchange) characters as DISPLAY files or in INTERNAL format. The DISPLAY file type, however, is more useful in most applications with this peripheral. DISPLAY is the default value and need not be specified.
- Open mode—This entry determines whether the file may be read from (INPUT), written to (OUTPUT), or both (UPDATE). If you do not specify an open mode, the default is UPDATE.

  Note: The RS232 port functions in all the above modes, but the parallel port must be opened in OUTPUT mode. Neither port may be opened in APPEND mode.
- Record type—*Records* are the blocks of data which make up file contents. The *HEX-BUS*™ interface system uses VARIABLE records whose length you can define. If no length is specified, the RS232 peripheral supplies an 80-character default. You can set maximum record length to either more or fewer than 80 characters by following the keyword VARIABLE with a number (as in the first example below). This feature is useful when, for instance, a printer having a line length of other than 80 characters is attached to the RS232 or parallel port.

Examples:

```
100 OPEN #1,"20.B=4800",OUTPUT,VARIABLE 132
```

Prepares a device on the RS232 port for operation. Certain software options or file attributes are specified while the rest (by being omitted from the OPEN statement) retain their default values. The peripheral is opened in DISPLAY format (by default) in OUTPUT mode. Record length is set to 132 characters. This length is appropriate for a device such as the TI 810 printer, which can print up to 132 characters per line.

```
110 OPEN #2,"50",OUTPUT
```

Enables a printer on the parallel port for operation. OUTPUT is specified because the parallel port must be opened in this mode. All other options are left at their default values.

The term *file* generally refers to a collection of data stored in a mass storage device such as the TI *Wafertape*™ peripheral. Although most devices accessible through the RS232 peripheral are display-type equipment not strictly oriented towards use with files, the word is employed in this manual for the sake of overall consistency and simplicity.

The CLOSE Statement

A device enabled for use with an OPEN statement must also be closed when its use is ended. The CLOSE statement has the following general form.

CLOSE #*file-number*

File-number is the number specified in the associated OPEN statement.

The INPUT Statement

When you use the INPUT statement with a device connected to the RS232 port, data can be received from the device and assigned to the variables listed in the statement. In this application, the INPUT statement has the following general form.

INPUT #*file-number, variable-list*

To use the INPUT statement with a device on the RS232 port you must open the device in INPUT or UPDATE mode. The amount of data taken depends on the transfer type and record length established in the associated OPEN statement. These constraints are detailed in the *Software Options* section of the manual, under the heading *Transfer Type.*

Examples:

```
200 INPUT #3,B$
```

Puts the next character string available from the device opened as #3 into the string variable B$.

```
210 INPUT #5,A,B,C
```

Puts the next three values from the device opened as #5 into variables A, B, and C.

See *The LINPUT Statement* for additional examples.

The LINPUT Statement

LINPUT accepts as a single string variable any data received during an input operation. The format for the LINPUT statement used with peripherals is as follows.

LINPUT *#file-number, string-variable*

In contrast to the INPUT statement, LINPUT is unaffected by punctuation or spaces in the data being accepted—information is stored exactly as received. The following table gives examples of the difference between the two statements.

DATA RECEIVED	PROGRAM SEGMENT	DATA DISPLAYED
"BROWN, CHARLES"	10 INPUT #1,A$ 20 DISPLAY A$	BROWN, CHARLES
"BROWN, CHARLES"	10 LINPUT #1,A$ 20 DISPLAY A$	"BROWN, CHARLES"
BROWN, CHARLES	10 INPUT #1,A$ 20 DISPLAY A$	BROWN
BROWN, CHARLES	10 LINPUT #1,A$ 20 DISPLAY A$	BROWN, CHARLES
CHARLES	10 INPUT #1,A$ 20 DISPLAY A$	CHARLES
CHARLES	10 LINPUT #1,A$ 20 DISPLAY A$	CHARLES

The data string "BROWN, CHARLES", put into memory by INPUT, is displayed as BROWN, CHARLES without quotation marks. When LINPUT is used, the display matches the original data. BROWN, CHARLES without quotation marks is taken by LINPUT as a single string variable. INPUT treats the comma as a separator and takes this data as two distinct variables.

When CHARLES, preceded by blank spaces, is entered, INPUT takes the blanks as zero input and disregards them. LINPUT takes them as actual input and reproduces the entire string, including the blanks.

As the examples show, LINPUT has great utility in situations involving transfers of data with punctuation, such as passages of text. In such applications you may often find it more useful than INPUT.

Example:

```
300 LINPUT #4,L$
```

Puts a record's length of data from the device opened as #4 into the string variable L$ (assuming a transfer type of T=R for "record"). Transfer type is discussed in the *Software Options* section of the manual.

Both INPUT and LINPUT take data from the RS232 port. When used without a file number, both statements also accept data from the computer console. Neither INPUT nor LINPUT, however, can be used with the parallel port.

The PRINT Statement

To send data from the computer through this peripheral to another device such as a printer or a display terminal, use the PRINT statement in either of the formats shown below. A parallel port device must first be opened in OUTPUT mode, and an RS232 device must be opened in OUTPUT or UPDATE mode.

PRINT #*file-number*, *print-list*

PRINT #*file-number*, USING{*string-expression* / *line-number*}, *print-list*

The optional USING clause specifies the printing or display format to be used. *String-expression* defines the format in the same way as the IMAGE statement (described in the BASIC reference manual). *Line-number* refers to the line number of an IMAGE statement. When used without a file number, both forms of the PRINT statement place data in the computer display.

Examples:

```
400 PRINT #9,A$
```

Sends the value of the variable A$ to the device opened as #9.

```
410 PRINT #9,"HELLO"
```

Sends the string HELLO to the device opened as #9.

```
420 PRINT #9, USING "###.#",357.97
```

Outputs the value 358.0 to the device opened as #9.

When data is transmitted through the RS232 peripheral to another device, a carriage return followed by a linefeed character is sent after each record unless *carriage return* software options N or C are set. For a description of the carriage return and linefeed functions, refer to the section on the *carriage return* software option.

The LIST Command

The lines of a program in the computer can be sent through this peripheral for display or printing on another device by use of the LIST command in the following format.

LIST "*device-number* [*.software-options*] " [, *line-list*]

No file number is needed; the LIST command opens and closes the device automatically. Using LIST without specifying a device number causes the program lines to be listed in the computer display.

Examples:

```
LIST "20.B=4800,N=10",10-100
```

Causes device 20 (attached to the RS232 port) to list lines 10 through 100 of a program stored in the computer. Software options needed for a match in device characteristics are specified. Those not specified remain at their default values.

```
LIST "50"
```

Causes an entire program to be printed through the parallel port. The software options are left at their default values.

The EOF Function

BASIC functions are program elements that return a value after manipulating data, based on parameters given in the function statements. You can use EOF (end-of-file) when accepting input from mass storage devices to test whether the end of a data file has been reached. With devices such as the *Wafertape*™ peripheral, the value returned by the EOF function (0 or –1) depends on where you are in the file being read. A –1 denotes the end of a file. A 0 means no end-of-file.

Since the purpose of the RS232 peripheral is not mass storage, EOF has little application with this device. The value returned is always 0, indicating that no end-of-file has been reached.

The CALL IO Statement

CALL IO is a BASIC statement used with the CC-40 computer to access special features and capabilities unique to many *HEX-BUS*™ peripherals.

The RS232 peripheral supports service requests. With this peripheral, the most frequent use of CALL IO—other than in the power-up check—is to enable attached devices to issue service requests or to disable them from doing so. An example of this CALL IO application can be found in appendix B.

For further information about the CALL IO statement, consult the manual for the Editor/Assembler software package.

Software Options

Devices that can be attached to this peripheral differ in how quickly they operate, how much data they can handle in a given time, and so on. You can compensate for such differences by selecting options which are available as part of the OPEN statement and the LIST command in BASIC. Because these options are set within programs, or software, they are known as *software options.*

The RS232 peripheral has 12 software options. Ten of these options affect the RS232 port, which is available for both input and output. The remaining two control the parallel port, which can be used only for output.

You can match the operating characteristics of this peripheral to those of other devices by indicating desired software options in the OPEN statement and in the LIST

command. Since the rest of the BASIC statements and commands that control this peripheral operate on previously opened files, you do not need to list options with any of them—only with OPEN and LIST.

The RS232 peripheral automatically assumes certain preset software options, called *default values.* You only have to specify those settings that you wish to change from their default values. The tables on the next two pages summarize the options, their formats, and their default values.

The software options are described in detail on the pages following the tables. Examples of their application occur in this section as well as in the COMMON APPLICATIONS section and in appendix B. In all examples the formats for specifying software options are those used with the CC-40.

SOFTWARE OPTIONS (RS232 PORT)

OPTION	DEFAULT	FORMAT	SETTINGS
Baud rate	300	B=	50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, 19200
Data bits	7	D=	*5, 6, 7, 8
Parity	O	P=	O, E, N, S, M
Parity check	N	C=	N, Y
Nulls	0	N=	0-99
Stop bits	1	S=	*1, 2
Echo	Y	E=	N, Y
Transfer type	R	T=	R, C, W
Data overrun	Y	O=	N, Y
Carriage return	L	R=	N, C, L

*If settings D=5 and S=2 are used, the actual number of stop bits is 1½.

Note: Several of the above options have equivalent options in the RS232 interface designed for the TI-99/4A Home Computer. Where such a correspondence exists, the RS232 software options listed above may be specified either in the CC-40 format shown or in the alternate 99/4A format. The RS232 options that have this dual-format feature are tabulated on the following page.

SOFTWARE OPTIONS (PARALLEL PORT)

OPTION	DEFAULT	FORMAT	SETTINGS
Carriage return	L	R=	N, C, L
Strobe level	N	S=	P, N

RS232 SOFTWARE OPTIONS WITH ALTERNATE FORMATS

OPTION	STANDARD FORMAT	ALTERNATE FORMAT	REMARKS
Baud rate	B=	.BA=	
Data bits	D=	.DA=	
Parity	P=	.PA=	
Parity check	C=	.CH	.CH is equivalent to C=Y. Tells the peripheral to check parity.
Nulls	N=	.NU	.NU is equivalent to N=6. Specifies six null characters.
Stop bits	S=	.TW	.TW is equivalent to S=2. Specifies two stop bits.
Echo	E=	.EC	.EC is equivalent to E=N. Turns the echo feature off.
Carriage return	R=	.CR	.CR is equivalent to R=N. No carriage return or linefeed is sent.
		.LF	.LF is equivalent to R=C. Specifies carriage return only—no line feed is sent.

Baud Rate

The *baud rate* is a measure of the speed of data transmission. After it has been set (in an OPEN statement) it remains that way unless reset in another OPEN statement, or until the peripheral is turned off. You can choose baud rates of 50, 75, 110, 135, 150, 300, 600, 1200, 1800, 2400, 3600, 4800, 7200, 9600, and 19200. The rate you choose should match the rate of the device being used. If you specify the baud rate as B = 1800, for example, the RS232 port is set up to transfer data at a rate of 1800 baud. The power-up default is 300. If you specify a different baud rate, it becomes the new default until the peripheral is turned off or a new baud rate is specified.

Data Bits

The *data bits* option lets you set the number of bits contained in each character transmitted. You have a choice of five, six, seven, or eight bits. If you specify D = 7, then seven-bit characters are sent. This is the ASCII standard and the power-up default value. If you set the number of data bits to 5, 6, or 8, this setting becomes the new default value until the peripheral is turned off or a different number is specified.

Parity

Checking parity is a way of detecting errors in data transmission. A *parity bit* can be inserted into each byte of data sent. It may be either 1 or 0, depending on the type of parity chosen. This peripheral can check certain types of parity at the RS232 port. If the type you specify in the OPEN statement is one of those that can be checked, any loss or distortion of data during transmission produces an error indication. The parity options are as follows.

- E for even parity.
- O for odd parity.
- S for space parity.
- M for mark parity.
- N for no parity.

As an example, specifying P = E for even parity sets the parity bit in a data byte to 0 if the byte has an even number of 1s and to 1 if the byte has an odd number of 1s. Odd parity is the power-up default. Once the parity is set, it remains as specified until changed by a new OPEN statement. The parity types are described in appendix C.

Parity Check

The peripheral can check for odd or even parity at the RS232 port. The *parity check* option allows you to control the use of this capability. C = N is the default value. When this setting is chosen or when no setting is specified, parity is not checked. Specifying C = Y in an OPEN statement enables the peripheral to monitor odd or even parity.

Nulls

The *nulls* option determines how many null characters follow a carriage return character. Nulls are included to give a printer time to complete its return before it begins printing characters. You may specify any number from 0 to 99. With the option N = 7, for instance, seven null characters are sent after a carriage return. The default value is N = 0.

Stop Bits

Stop bits are sent after transmission of each character to indicate the end of that character. With the *stop bits* software option, you can set the number of stop bits to either 1 or 2. The setting S = 2, for example, causes this peripheral to end each transmission with two stop bits. The default value is S = 1. Note that if the settings D = 5 and S = 2 are used, the actual number of stop bits is 1½.

Echo

The *echo* software option provides an additional method of detecting errors in transmission of data from a peripheral device through the RS232 port to the computer. The RS232 peripheral can be instructed to retransmit characters, as it receives them, back to the sending device. You can select E = Y to have characters echoed back to the sending device or E = N to prevent them from being echoed back. The default is E = Y.

Transfer Type

Incoming data is temporarily stored at the RS232 port. The *transfer type* software option determines the form in which data is moved from the port to the computer.

Option T = R, the default setting, sends input to the computer in records of the length established in the associated OPEN statement. If VARIABLE 15 is specified, for example, data is transferred in records 15 characters long. This feature is useful in applications that involve data received in records of some consistent length.

Should a carriage return end the data stream before a record is complete, the record goes to the computer with the remainder of its length padded by blank spaces. An entire record's length of blanks is sent to the computer if a carriage return is the first character input with option T = R.

If the data transmission itself has a series of trailing blanks between the carriage return and the end of the actual data, the user's choice of input statement determines how the blanks are handled. INPUT removes trailing blanks. LINPUT preserves them as part of the data.

Option T = C transfers data character by character. Within limits, any characters stored in the peripheral at the time of an input operation (including carriage returns and linefeeds) are sent to the computer. The limits are set by the record length specified in the OPEN statement and by the number of characters in the peripheral: the lower of these two figures is the number of characters transferred. If no data is available, a null data string is sent to the computer and the program continues.

Option T = W also transfers data character by character. As long as data is in the peripheral there is no difference between this option and option T = C, but when no data is available the RS232 port causes the computer to wait until a character is ready.

Setting the transfer type to T = C, for example, allows sending as many characters as are stored at the port, up to the specified record length, to the computer.

Data Overrun

A data overrun occurs when the RS232 port receives more data than it can send on to the computer. Any additional data that arrives during an overrun is lost. Use the *data overrun* option to specify whether you wish the peripheral to report data overrun as an error.

Choosing the option O = N causes the peripheral to disregard data overruns. When you are receiving data through the RS232 port from a device such as a video terminal, this setting allows the incoming data stream to overflow the port without producing an error indication and stopping the program. Characters lost during an overrun are not echoed back to the sending device, regardless of the *echo* option setting.

Selecting the option O = Y (which is the default value) allows a data overrun error to stop a program. This feature is helpful when the loss of even a small amount of data would make the rest of a transmission useless.

Carriage Return

The *carriage return* option allows the peripheral to regulate the carriage operation of a printing or display device connected to the RS232 port. The peripheral can be directed either to send an automatic linefeed and carriage return (or just a carriage return) at the end of each record, or to suppress those characters.

The carriage return sends the print head or cursor of the attached device back to the starting point. The linefeed character causes the paper or display to advance by one line. The options are as follows.

- R = L for carriage return and linefeed.
- R = C for carriage return only.
- R = N for no carriage return or linefeed.

Selecting the option R = C, for example, means that each record sent from the peripheral is followed by a carriage return but not by a linefeed. The default value is L. These details apply equally to the parallel port *carriage return* option, which has the same selections and default.

Carriage Return (Parallel Port)

The *carriage return* option for the parallel port works in the same way as the *carriage return* option for the RS232 port. It determines whether the peripheral sends or suppresses an automatic linefeed and carriage return (or carriage return only) at the end of each record. The options and default are identical to those of the RS232 port.

Strobe Level (Parallel Port)

A strobe is a control signal used in data transfers. Some printers which can be connected to the parallel port require positive strobes, while others use negative strobes (both described in appendix E). The *strobe level* software option lets you select which level to use with a given printer. The setting S = P in an OPEN statement produces a positive strobe. S = N (which is the default value) produces a negative strobe. Consult the user's manual of the printer you intend to connect for the proper strobe level.

RECOVERY FROM ERRORS

Most errors that involve the RS232 peripheral occur during input or output with this device as a result of some discrepancy in program statements. These errors are seldom serious, and they are easily remedied. Errors are normally indicated to the operator in the form of a displayed message beginning `I/O error...` which includes a one- to three-digit error code and the device or file number of the port concerned. If the error occurs during execution of a program, any open files are automatically closed. The error codes that you may encounter with this peripheral are listed in appendix A with their meanings and recommended corrective steps.

In the event of such an error, first note the code number and look it up in appendix A. Next clear the fault indication and restore the cursor by pressing **[CLR]**. Then try the operation again, following the suggestions given in the appendix for that error.

If the above measures do not solve the problem, turn the peripheral off briefly and try once more. If the malfunction persists, turn the computer off momentarily and try again. Successive attempts that produce the same error may indicate a hardware fault. Consult the *In Case of Difficulty* section of this manual for service information and additional troubleshooting suggestions.

COMMON APPLICATIONS

The RS232 peripheral lets you operate your computer with a wide range of devices beyond those in the standard group of *HEX-BUS*™ peripherals. Two examples are given here. The first shows how data can be sent from the computer to a printer on the parallel port, and the second illustrates how you can use the computer to communicate with a video display terminal on the RS232 port.

Output to a Printer on the Parallel Port

In this example, seven character strings are read from data statements at the end of the program and printed.

```
100 DIM A$(7)
200 FOR P=1 TO 7:READ A$(P)
300 NEXT P
400 OPEN #23,"50",OUTPUT
500 FOR S=1 TO 7:PRINT #23,A$(S)
600 NEXT S
700 CLOSE #23
800 DATA "MONDAY","TUESDAY","WEDNESDAY","THURSDAY"
900 DATA "FRIDAY", "SATURDAY","SUNDAY"
```

Lines 100-300 set up an array and read in the data. Line 400 assigns file number 23 to device number 50, a printer on the parallel port. The port is opened for OUTPUT. Because no file type is specified, DISPLAY is assigned by default and the attached printing device receives data in ASCII format. *Carriage return* setting L, also selected automatically, sends both a carriage return and a linefeed character to the printer at the end of each record. In lines 500 and 600 a PRINT loop is executed seven times. Each character string is printed on a separate line.

If the printer is off-line when the PRINT loop is executed, the RS232 peripheral holds the data ready and waits. After all the lines are printed, the device is closed.

Exchanging Messages Between a Computer and a Video Terminal

The following program can be used in a *HEX-BUS*™ system with a video display terminal connected to the RS232 port. The program first accepts a message from the computer and

transmits it to the terminal. The terminal user is then prompted to enter a message which is sent to the computer display. The exchange of messages can continue for as long as the program is run.

```
100 OPEN #1,"20.B=4800,O=N,R=N,C=Y,P=0,D=7",
    DISPLAY, UPDATE
200 INPUT "TYPE MESSAGE: ";X$
300 PRINT #1,"MESSAGE FROM COMPUTER:
    ";X$;CHR$(10);CHR$(13)
400 PRINT #1,"TYPE MESSAGE: "
500 INPUT #1,X$
600 PRINT #1,CHR$(10)
700 PRINT "MESSAGE FROM TERMINAL: ";X$
800 PAUSE
900 GOTO 200
```

Line 100 opens file number 1 for the RS232 port and defines a number of parameters for operation—baud rate, management of data overruns, carriage return and linefeed setting, parity check, parity type, number of data bits, file type, and open mode. More details on these parameters can be found in the USING BASIC section. They are set up to match the operating characteristics of the RS232 peripheral to those of the terminal.

Line 200 prompts the computer user to type a message and puts it into the string variable X$. Line 300 sends the message to the terminal, where it is displayed with the introductory words MESSAGE FROM COMPUTER:. Both of these items are displayed on the same line.

In lines 400 and 500, the terminal user is prompted to type a message which is sent back to the computer. So that the terminal user's prompt and message do not appear on the same line as the computer user's, carriage return and linefeed characters—CHR$(13) and CHR$(10)—are inserted after the message from the computer in line 300.

In line 600, a linefeed character is sent to the terminal so that the next message from the computer does not erase the terminal user's current message from his display. Lines 700 and 800 place the terminal user's message in the computer display. The PAUSE instruction holds the message in the computer display until the computer user presses the **[ENTER]** key to resume the exchange.

APPENDIX A: ERROR CODES

Listed below are the error codes related to the operation of the RS232 peripheral in BASIC programs.

The error codes marked with an asterisk (*) are those that apply only if you use the CALL IO instruction. They do not occur with any other BASIC statements or commands.

CODE	MEANING
0*	NO ERRORS.
1	DEVICE/FILE OPTIONS ERROR. Check the software options in the OPEN statement or IO call. Make sure that commas, periods and equal signs are used correctly.
2	ERROR IN ATTRIBUTES. Check the file attributes given in the OPEN statement or IO call.
4	DEVICE/FILE NOT OPEN. Open the device before using it.
5	DEVICE/FILE ALREADY OPEN. Close the device and try again. If the CLOSE statement in BASIC does not work, use **CALL IO(20,1)** to close the RS232 port or **CALL IO(50,1)** to close the parallel port. If the above steps are not successful, you can close all peripherals by turning the computer off momentarily.
8*	DATA/FILE TOO LONG. Modify the Peripheral Access Block.
10*	NOT REQUESTING SERVICE. This message may be sent by a peripheral in response to a poll by the computer to determine which device issued a service request.
12*	BUFFER SIZE ERROR. The data buffer size specified in the Input/Output subsystem instructions is not large enough for the data returned by a peripheral. Make the buffer larger.
13	UNSUPPORTED COMMAND. The peripheral generates this message in response to commands that it cannot accept.
14	DEVICE/FILE NOT OPEN FOR OUTPUT. The peripheral's current open mode does not allow you to send data. Reopen the device in OUTPUT (either port) or UPDATE (RS232 port) mode.

15 DEVICE/FILE NOT OPEN FOR INPUT. The peripheral's current open mode does not allow you to receive data. Reopen the RS232 port in INPUT or UPDATE mode.

17 RELATIVE FILES NOT SUPPORTED. The RS232 peripheral can only process SEQUENTIAL files.

19 APPEND MODE NOT SUPPORTED. This peripheral cannot operate in the APPEND mode. It must be opened in INPUT, OUTPUT, or UPDATE mode, depending on its application in a given program.

21 INPUT MODE NOT SUPPORTED. The peripheral returns this message if you try to open the parallel port in the INPUT mode. The port is provided specifically for output to printers. It must be opened in the OUTPUT mode.

22 UPDATE MODE NOT SUPPORTED. The peripheral returns this message if you try to open the parallel port in the UPDATE mode. Use the OUTPUT mode.

80 DATA OVERRUN. Data is arriving at the RS232 port faster than it can be sent on to the computer. Input more frequently to save data, or set the *overrun* software option to O = N to ignore the loss of data.

81 PARITY ERROR. Indicates either that a data transmission error has occurred or that the parity settings of the RS232 port and the attached device do not match. Check that the parity settings are the same.

82 FRAMING ERROR. A disparity exists between the *data bits, stop bits, baud rate,* or *parity* software option settings of the RS232 port and the characteristics of the attached device. Be sure that these settings match the device characteristics.

83 FRAMING AND PARITY ERRORS. A combination of errors 81 and 82 is occurring. Correct the problem as described above.

255 TIME-OUT ERROR. The computer generates this error code if it cannot communicate with a peripheral. Check the *HEX-BUS* cable connections and make sure that you are using the correct *device-number.*

APPENDIX B: USING ASSEMBLY LANGUAGE

This appendix is primarily for use in conjunction with the Editor/Assembler manual and software package created for the CC-40 computer and its peripherals. The descriptions and sample application given here supplement the explanations and examples found in the Editor/Assembler manual.

The CC-40 has an Input/Output subsystem which you can access to communicate with *HEX-BUS*™ peripherals on an assembly-language level. This communication can take place either entirely in assembly language if you have the Editor/Assembler cartridge, or from BASIC if you use the CALL IO instruction with appropriate command codes.

Command Codes

The devices that you can attach to the RS232 peripheral are essentially display-type devices, not oriented toward use with files. The commands affecting this peripheral involve the basic functions of opening or closing devices and reading or writing data. (The RS232 port also supports service requests.) The assembly-language command codes with which you can direct the RS232 peripheral to carry out these tasks are listed below, together with any applicable restrictions.

0 *Open.* Prepares a device for use. The APPEND mode and RELATIVE file type cannot be used with either port of the peripheral. The UPDATE and INPUT modes can be used only with the RS232 port.

1 *Close.* Completes any pending operations by a device and ends the use of the device until the next *open* instruction.

3 *Read.* Inputs data from a peripheral device. The *read* command can be used with the RS232 port, but not with the parallel port. The attached device must first be opened in INPUT or UPDATE mode.

4 *Write.* Sends data to a peripheral device. The *write* command can be used with both ports of this peripheral. The parallel port must first be opened in OUTPUT mode and the RS232 port in OUTPUT or UPDATE mode.

7 *Return Status.* Used in requesting device and file status information from peripherals. When this message is used with the RS232 peripheral, the *end-of-file* flag is always returned set to zero.

8 *Enable Service Requests.* Signals a peripheral device that it may issue service requests to the computer. This command is used only with the RS232 port since the parallel port cannot send messages to the computer. The next section of this appendix contains further details.

9 *Disable Service Requests.* Signals a peripheral device that it may no longer request service. This instruction can be used only with the RS232 port.

10 *Service Request Poll.* Not used in programming, but automatically sent to peripherals by the computer upon reception of a service request. Determines which device is requesting service.

15 *Set Options.* Modifies the peripheral options previously specified in an *open* instruction without first closing and reopening the device.

16 *Transmit Break.* Causes the RS232 peripheral to send a continuous break signal for approximately .25 seconds. Used in data communications to shift between operating modes.

254 *Null Operation.* Not used in programming, but automatically sent by the computer in response to any service requests received while a previous request is being processed. Though the computer may enable a number of peripherals for service requests, it can process only one request at a time.

255 *Reset Bus.* Not normally included in programs, but allows the user to close all open device files and reset all peripherals attached to the computer.

For a detailed treatment of the Input/Output subsystem and command codes, refer to the manual for the Editor/Assembler cartridge.

Service Requests

Many *HEX-BUS*™ peripherals have unique capabilities which are accessible on an assembly-language level. The RS232 peripheral can be enabled for service requests. It then has the ability to interrupt regular program execution and have data processed by the computer.

This peripheral, when enabled, issues a service request to the computer whenever data from the attached device appears at the RS232 port. The peripheral is enabled for service requests as follows.

1. Set up a Service Request Peripheral Access Block (SRPAB).
2. Enable interrupts.
3. Open the device.
4. Send the *enable service requests* instruction.

The above steps are executed most efficiently in assembly language, as described in the Editor/Assembler manual. With somewhat reduced efficiency, they can also be performed in BASIC. Since assembly-language programming is not within the scope of this manual, a BASIC example is given here.

The SRPAB, set up in computer memory by the user, supplies the parameters necessary for communication with the RS232 peripheral through the Input/Output subsystem. It contains the one- or two-byte fields diagrammed below, which are filled as needed for each particular application. The diagram shows an SRPAB that allows service requests from a device on the RS232 port.

	SRPAB$	
Device number	20	RS232 port
Command code	0	Don't care
Logical unit number (LUNO)	0	Don't care
Record number	0	Don't care
	0	
Buffer length	10	Size of allocated buffer
	0	
Data length	0	Not sending any data
	0	
Returned status	0	Zeroed initially
Buffer pointer	LSB	Points to highest address
	MSB	in buffer
Link to next SRPAB	LSB2	Points to start of this SRPAB
	MSB2	(No other device enabled)
Service flag	0	Zeroed initially
Pointer to device service	0	Left zeroed
routine (DSR)	0	

As the diagram indicates, the elements that must be specified in building a Service Request PAB for the RS232 peripheral are the device number, buffer length, data length, return status, buffer address, SRPAB pointer, service flag, and device service routine (DSR) pointer. The field for the DSR pointer is provided for future applications. It is not used in the CC-40 and should be left zeroed as shown.

Data is entered in the fields of an SRPAB with the POKE subprogram. The necessary values are placed in memory starting at the lowest address allocated. The following program segment illustrates the four-stage process of enabling the RS232 peripheral for service requests, beginning with the construction of an SRPAB.

```
100 CALL GETMEM(17,SRPABADDR) !Gets SRPAB
110 CALL GETMEM(10,BUFFADDR) !Gets buffer
120 CALL SPLIT (BUFFADDR+9,MSB,LSB)!Builds SRPAB
130 CALL SPLIT (SRPABADDR+16,MSB2,LSB2)
140 CALL POKE (SRPABADDR,0,0,0,MSB2,LSB2,MSB,LSB,0,
    0,0,0,10,0,0,0,0,20)
150 CALL POKE (2056,MSB2,LSB2) !Sets up pointer to
    SRPAB
160 CALL POKE (BUFFADDR,5,10) !Enables interrupts
170 CALL EXEC (BUFFADDR)
180 CALL PEEK (256,IOCNTRL)
190 IOCNTRL=IOCNTRL OR 1
200 CALL POKE (256,IOCNTRL)
210 OPEN #1,"20.B=9600,C=Y,P=E,D=7,E=Y,T=C",
    VARIABLE 10
220 CALL IO(20,8) !Sends enable service requests
    command
```

Lines 100-140 above prepare memory space for the SRPAB and the data buffer. Line 150 sets up a pointer to the SRPAB for the I/O subsystem. In lines 160-200, interrupts are enabled so that the computer can recognize service requests. In line 210 the peripheral is opened, as required before enabling service requests. The *enable service requests* command is sent in line 220.

The SPLIT subprogram which is called in lines 120 and 130 splits a 16-bit unsigned data value into a Most Significant Byte (MSB) and a Least Significant Byte (LSB), as shown below.

```
1000 SUB SPLIT (DATAVAL,MSB,LSB)
1010 MSB=INT(DATAVAL/256)
1020 LSB=DATAVAL-MSB*256
1030 SUBEND
```

After a device is enabled for service requests, a program can check for received data by testing the service flag in the SRPAB as shown below. The I/O subsystem sets the service flag upon completion of a successful poll.

```
230 CALL PEEK (SRPABADDR+2,SERVICE_FLAG)
240 IF SERVICE_FLAG<>0 THEN 700
250 GOTO 230
```

If the service flag check indicates no reception of data, the main program continues immediately after line 240. If the service flag is set, however, execution of the main program halts and the program branches to line 700. At line 700 the operation status is checked. Data is then removed and processed.

```
700 CALL PEEK (SRPABADDR+7,STATUS)
710 IF STATUS THEN PRINT
"Error--";STATUS:PAUSE:STOP
720 CALL PEEK (SRPABADDR+8,MSB,LSB)!Gets data length
730 DATLEN=MSB*256+LSB !MSB should always be zero
740 DATA$="" !Initializes data string
750 FOR I=1 TO DATLEN
760 CALL PEEK (BUFFADDR+10-I,A) !Gets next character
770 DATA$=DATA$&CHR$(A) !Appends it to the string
780 NEXT I
790 PRINT DATA$
800 PAUSE
```

The number of characters transferred during each service request depends on the buffer length and transfer mode established in the OPEN command. If the option T = R is specified, then a full record of data (or data padded with blanks) having the length of the buffer is sent. If the record is not complete when the computer polls the RS232 port, the peripheral causes the computer to wait until it is complete. If the options T = C or T = W are chosen, any number of characters up to the buffer length may be transferred.

Until the SRPAB service flag is reset to zero by the program, any additional service requests are disregarded. The peripheral requesting service continues doing so until its request is acknowledged.

If the computer transmits an input instruction to a peripheral issuing a service request, the service request is terminated and the peripheral sends the required data instead. Data can also be output to devices enabled for service requests. These operations are executed as usual.

APPENDIX C: PARITY OPTIONS

Listed and defined below are the parity options available to you in data transmissions with the RS232 peripheral. This peripheral checks for even and odd parity.

E *Even parity.* If a data byte in the transmission contains an odd number of 1s, its parity bit becomes 1. If the data byte has an even number of 1s, its parity bit becomes 0.

O *Odd parity.* If a data byte in the transmission has an even number of 1s, its parity bit is set to 1. Otherwise, its parity bit is set to 0.

S *Space parity.* If you specify space parity, the parity bit is always 0 regardless of whether the data byte contains an even or odd number of 1s.

M *Mark parity.* The parity bit is always 1.

N *No parity.* The data transmission does not contain a parity bit.

APPENDIX D: RS232 CABLING

Cables used to connect other computers, calculators, modems, or terminals to the RS232 port must have EIA RS232C 25-pin male connectors. Some electronic devices built according to the RS232C standard use nearly all the available lines, while other devices require fewer connections. This peripheral utilizes the nine pins listed below.

PIN	SIGNAL	DIRECTION
1	Frame ground	—————
2	Serial data in	To HX-3000 RS232 port
3	Serial data out	From HX-3000 RS232 port
4	Request to send	To HX-3000 RS232 port
5	Clear to send	From HX-3000 RS232 port
6	Data set ready	From HX-3000 RS232 port
7	Signal ground	From HX-3000 RS232 port
8	Data carrier detect	From HX-3000 RS232 port
20	Data terminal ready	To HX-3000 RS232 port

The RS232C peripheral connects directly to most video display terminals and serial printers such as the Texas Instruments 99/4 Impact Printer without modification. In these applications the following lines are used.

HX-3000 RS232 PORT	DATA TERMINAL EQUIPMENT
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
20	20

The Texas Instruments Telephone Coupler (modem) is compatible with this peripheral in all respects. Other modems may require cabling that differs slightly from the configuration shown. If you use a modem other than TI's, check its manual to determine whether any wiring changes are necessary for compatibility. In general, other devices such as modems require the pin connections listed below. Pin 4 is jumpered to pin 20 inside the RS232 peripheral.

HX-3000 RS232 PORT	DATA COMMUNICATIONS EQUIPMENT
1	1
2	3
3	2
6	20
7	7
8	8
20	6

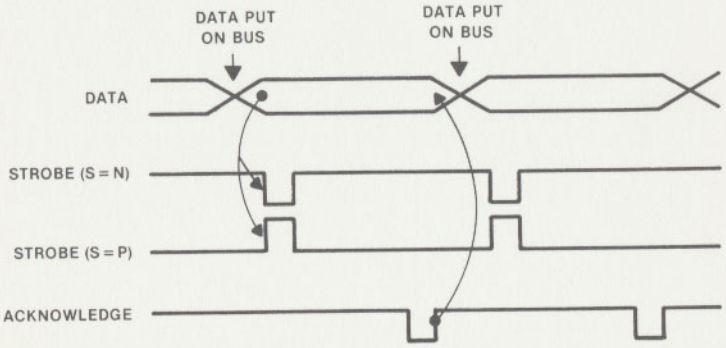
APPENDIX E: PARALLEL PORT CABLING

Peripherals equipped with the parallel port option have a permanently-attached flat cable which is fitted with the Amphenol 36-pin male type of connector or its equivalent. These are standard parallel connectors. Twenty of the available pins are used. Eight lines transfer data, while two others carry control signals. The remaining 10 lines serve as signal grounds. The pins are connected as listed below.

PIN	FUNCTION
1	Strobe
2	D0 (least significant data bit)
3	D1
4	D2
5	D3
6	D4
7	D5
8	D6
9	D7 (most significant data bit)
10	Acknowledge
19-28	Ground

The signals on pins 1 through 9 are outputs from the parallel port to the printer. *Acknowledge* is an input to the parallel port from the printer.

The following diagram illustrates the sequence of events that occur during transmission of data from the parallel port. When a data byte appears at the port, the strobe (either positive or negative, as selected by the user) signals the attached printer that data is available. If the printer is ready to receive it, the data is transferred and processed. When the printer is ready to accept another byte, the *acknowledge* signal is sent back to the parallel port.



SERVICE INFORMATION

In Case of Difficulty

If this peripheral or attached devices do not appear to be working properly, check the following.

1. *Power*—Be sure that the power source is in order, the peripheral is plugged in, and the power switch is on. All peripherals must be turned on for proper operation.
2. When connecting devices to the HX-3000 RS232 port, be sure the correct cable is being used. Check for loose or broken leads and connectors. Be sure that cables are plugged in securely. Some devices may require special cable connections (see appendix D).
3. *Software Options*—Even when all devices are operating correctly, this peripheral can appear to malfunction if software options are improperly set. Check the user's manual of the device attached to the peripheral and verify that the device characteristics correspond to the options selected in the OPEN statement or LIST command. A printer built to operate at 300 baud, for instance, will not function if a baud rate of 1200 is chosen. In communications with other computers or with terminals, unsuitable *baud rate, data bits, parity, echo,* or *carriage return* settings may produce undesirable results.
4. *Attached Devices*—If a device has a test or local mode, use it to be sure that the device is working properly when disconnected from the HX-3000 RS232 peripheral.

If the RS232 peripheral or attached devices still do not appear to be working properly, first turn all power off. Next, disconnect this peripheral from the *HEX-BUS*™ interface and attached devices. Then follow the steps below.

1. *See if the computer itself is working properly.* Turn on the computer. Enter the statement **OPEN #1, "20"** (or **OPEN #1, "50"** if the malfunction is associated with the parallel port). The error message `I/O error 255 #1` should appear in the computer display, indicating that the port named cannot be opened. This result is expected when the RS232 peripheral is not connected to the *HEX-BUS* interface.

2. *Check that the RS232 peripheral is working properly.* Refer to the set-up instructions and reconnect only the HX-3000 RS232 peripheral to the *HEX-BUS* interface. Type the statement **OPEN #1, "20"** or **OPEN #1, "50"** and press **[ENTER]**. The OPEN statement should disappear from the computer display and be replaced by a flashing cursor on the left side, indicating that the port named has been opened.
3. If a device still does not work when reattached to this peripheral, then that unit or its cable may be faulty. Check the user's manual of the device for additional troubleshooting suggestions.
4. If none of the above procedures correct the difficulty, consult the section entitled *If You Have Questions or Need Assistance* or refer to the *Service Information* portion of the *User's Reference Guide.*

Exchange Centers

If your HX-3000 RS232 peripheral requires service and you do not wish to return the unit to a service facility for repair or replacement, you may elect to exchange the unit for a factory-reconditioned HX-3000 RS232 peripheral of the same model (or equivalent model specified by TI) by going in person to one of the exchange centers which have been established across the United States. A handling fee will be charged by the exchange center for in-warranty exchanges of the HX-3000 RS232 peripheral. Out-of-warranty exchanges will be charged at the rates in effect at the time of the exchange. Please refer to the enclosed Exchange Service listing or call the Consumer Relations Department for exchange fee information and the location of the nearest exchange center.

If You Have Questions or Need Assistance

If you have questions concerning HX-3000 RS232 peripheral repair or peripheral, accessory, or software purchase, please call our Customer Relations Department at (800) 858-4565 (toll free within the contiguous United States). The operators at these numbers cannot provide technical assistance.

For technical questions such as programming, specific applications, etc., you can call (806) 741-2663. Please note that this is not a toll-free number and collect calls cannot be accepted.

As an alternative you can write to

Consumer Relations Department
Texas Instruments Incorporated
P.O. Box 53
Lubbock, Texas 79408

Because of the number of suggestions which come to Texas Instruments from many sources containing both new and old ideas, Texas Instruments will consider such suggestions only if they are freely given to Texas Instruments. It is the policy of Texas Instruments to refuse to receive any suggestions in confidence. Therefore, if you wish to share your suggestions with Texas Instruments or if you wish us to review any BASIC language program which you have developed, please include the following statement in your letter.

> "All of the information forwarded herewith is presented to Texas Instruments on a nonconfidential, nonobligatory basis; no relationship, confidential or otherwise, expressed or implied, is established with Texas Instruments by this presentation. Texas Instruments may use, copyright, distribute, publish, reproduce, or dispose of the information in any way without compensation to me."

THREE-MONTH LIMITED WARRANTY

THIS TEXAS INSTRUMENTS HX-3000 RS232 PERIPHERAL WARRANTY EXTENDS TO THE ORIGINAL CONSUMER PURCHASER OF THE ACCESSORY.

WARRANTY DURATION

This HX-3000 RS232 peripheral is warranted for a period of three (3) months from the date of the original purchase by the consumer.

WARRANTY COVERAGE

This HX-3000 RS232 peripheral is warranted against defective materials or workmanship. **THIS WARRANTY IS VOID IF THE ACCESSORY HAS BEEN DAMAGED BY ACCIDENT, UNREASONABLE USE, NEGLECT, IMPROPER SERVICE OR OTHER CAUSES NOT ARISING OUT OF DEFECTS IN MATERIALS OR WORKMANSHIP.**

WARRANTY DISCLAIMERS

ANY IMPLIED WARRANTIES ARISING OUT OF THIS SALE, INCLUDING BUT NOT LIMITED TO THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO THE ABOVE THREE-MONTH PERIOD. TEXAS INSTRUMENTS SHALL NOT BE LIABLE FOR LOSS OF USE OF THE HX-3000 RS232 PERIPHERAL OR OTHER INCIDENTAL OR CONSEQUENTIAL COSTS, EXPENSES, OR DAMAGES INCURRED BY THE CONSUMER OR ANY OTHER USER.

Some states do not allow the exclusion or limitation of implied warranties or consequential damages, so the above limitations or exclusions may not apply to you in those states.

LEGAL REMEDIES

This warranty gives you specific legal rights, and you may also have other rights that vary from state to state.

WARRANTY PERFORMANCE

During the above three-month warranty period, your HX-3000 RS232 peripheral will be repaired or replaced with a new or reconditioned unit of the same or equivalent model (at TI's option) when the unit is returned by prepaid shipment to a Texas Instruments Service Facility listed below. The repaired or replacement unit will be warranted for three months from date of repair or replacement. Other than the postage requirement, no charge will be made for the repair or replacement of in-warranty units.

Texas Instruments strongly recommends that you insure the unit for value, prior to shipment.

TEXAS INSTRUMENTS CONSUMER SERVICE FACILITIES

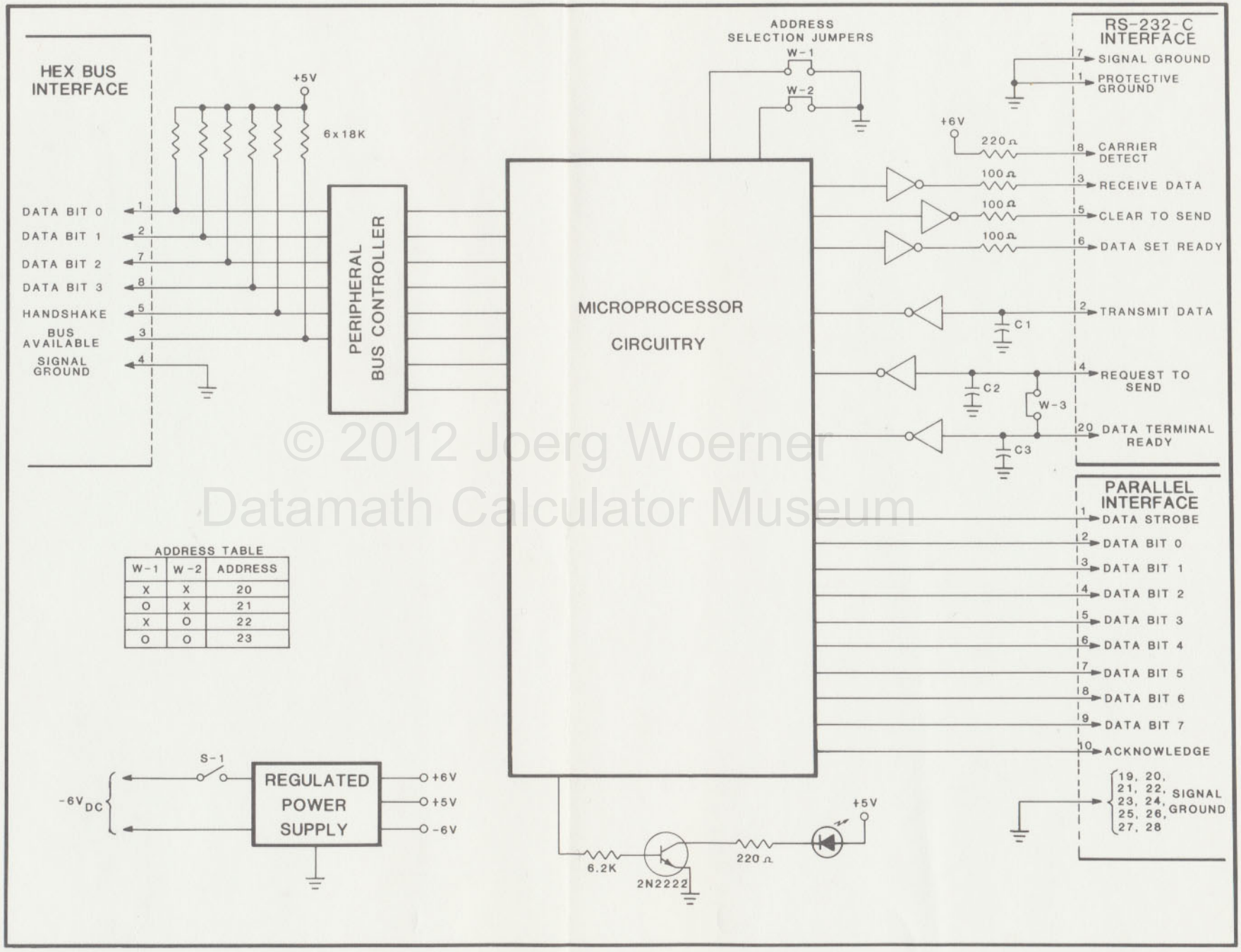
U.S. Residents
Texas Instruments Service Facility
2303 North University
Lubbock, Texas 79415

Canadian Residents
Geophysical Services Incorporated
41 Shelley Road
Richmond Hill, Ontario, Canada L4C5G4

Consumers in California and Oregon may contact the following Texas Instruments offices for additional assistance or information.

Texas Instruments Consumer Service
831 South Douglas Street
El Segundo, California 90245
(213) 973-1803

Texas Instruments Consumer Service
6700 Southwest 105th Street
Kristin Square
Suite 110
Beaverton, Oregon 97005
(503) 643-6758



Federal Communications Commission Requirements Concerning Radio-Frequency Interference

The Texas Instruments Compact Computer 40 and peripherals generate and use radio-frequency (RF) energy. If not installed and used properly (as outlined in the instructions provided by Texas Instruments), this equipment may cause interference to radio and television reception.

This equipment has been type-tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules. These rules are designed to provide reasonable protection against radio and television interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause interference to radio or television reception (which you can determine by turning the equipment off and on), try to correct the interference by one or more of the following measures.

- Reorient the receiving antenna (that is, the antenna for the radio or television that is "receiving" the interference).
- Change the position of the computer with respect to the radio or television equipment that is receiving interference.
- Move the computer away from the equipment that is receiving interference.
- Plug the computer into a different wall outlet so that the computer and the equipment receiving interference are on different branch circuits.

If these measures do not eliminate the interference, please consult your dealer or an experienced radio/television technician for additional suggestions. Also, the Federal Communications Commission has prepared a helpful booklet, "How to Identify and Resolve Radio-TV Interference Problems." This book is available from

The US Government Printing Office
Washington, D.C. 20402

Please specify Stock Number 004-000-00345-4 when ordering copies.

WARNING: This equipment has been certified to comply with the limits for a Class B computing device, pursuant to Subpart J of Part 15 of FCC Rules. Only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to the computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

TEXAS INSTRUMENTS
INCORPORATED
Dallas, Texas

Printed in U S A

1055945-1